COMPUTER-AIDED CONTROL SYSTEM ENGINEERING TOOLS

Christian Schmid
Institute of Automation and Computer Control, Ruhr-Universität Bochum, Germany

Keywords: Computer-aided control system design, modeling, multidisciplinary modeling, simulation, numerics, automatic code generation

Contents

Summary

Tools for computer-aided control system design have become available. With varying degrees of sophistication, all tools aid the control system engineer with efficient numerical analysis and graphical data representation. Tools, however, also go beyond theoretical control system analysis to offer support for the task of implementing control laws in hardware and software. This contribution starts with a historical review of control and the techniques in this area. The main part deals with techniques used by CACSD tools. The basic CACSD cycles, numerical aspects for computation, software libraries, modeling and interaction are in the focus. Furthermore, the dominant role of MATLAB is emphasized and trends are shown.

1. Introduction

Modern control techniques are based on more sophisticated process models, controller design and simulation studies. With varying degrees of sophistication computer-aided tools aid the control system engineer with sufficient numerical analysis and graphical data representation. This contribution deals with such tools for computer-aided design (CAD) exclusively in the context of control engineering. This term should emphasize the much broader facets of the problem, which the control engineer faces, than the common term computer-aided control system design (CACSD). Especially, the aspects of modeling, analysis, and simulation must always be considered in addition to design.

Therefore computer-aided design in this context can be defined as the use of digital computers as primary tool during the modeling, identification, analysis, design and implementation phase of control engineering.

All of these aspects had not been always in the focus of the research in this area. The development has partly gone in pace with the rapid advances of digital computer technology and together with new software engineering techniques.

1.1. Historical Background in Control Engineering

In the 1950s and 1960s of the last century engineers were dealing with isolated low-order continuous-time systems (see *Description of Continuous Linear Time-Invariant Systems in Time-Domain* and *Description of Continuous Linear Time-Invariant Systems in Frequency Domain*). The design of these systems was done in the frequency domain and most of the techniques developed were graphical in nature. The tools and the methods were simple enough that the engineer could easily master them in a relatively short time. With the need of dealing with more complex systems with multiple inputs and outputs, these graphical techniques failed to provide sufficient insight. When engineers tried to introduce automatic control methods into e.g. power stations, complex systems had to be modeled in the time domain.

From the early 1960s several computerized methods, which support modeling and transformation techniques between frequency and time domain demonstrates this. From that time some digital computer applications are known. It was the state-space approach (see *Description and Analysis of Dynamic Systems in State Space*), which opened the way for dealing with more complex systems in a systematic manner. This technology of the 1960s furnished most of the matrix-based design methods we have today. Lately, computers have given the traditional frequency-domain methods (see *Design in Frequency Domain*) a renaissance through the use of new, computer-based algorithms for multivariable controller design (see *Controller Design*). The control theory has evolved to a state where the digital computer has become a requirement for the control system engineer, and computer-aided control system design has emerged as an indispensable tool.

1.2. Historical Background in First-generation CAD Tools

In the 1960s and early 1970s, researchers concerned themselves mostly with the efficient design of control algorithms. It was felt that the numerical algorithms that were needed to analyze systems and to design controllers were quite delicate, whereas the resulting control algorithm themselves and the implementation of these control strategies, once designed, were fairly trivial undertakings. The research efforts resulted in a set of more or less decently maintained libraries of subroutines. The users of these subroutines concerned themselves only with the subroutine interfaces, and had often little to no understanding of the computations that went on inside these routines.

The availability of reliable computer systems in the late 1970s led to the idea to realize the complete development cycles for control system design. The tasks to be performed were from modeling until to the final implementation in a digital controller. At that time, the early idea of the later “fast prototyping” in control system design was born. Several examples of such full-cycle CAD systems are known from literature. They were relatively large, more or less modular systems offering many different features. The main characteristics of such systems are that the same software allows performing all phases in the design cycle of control systems.

This includes data acquisition, signal processing, modeling, identification, model validation, analysis, controller design, simulation and finally real-time implementation (see *Real-time Implementation*). These systems had been developed and applied until the 1980s. But this first generation of tools had offered monolithic building blocks with too coarse a granularity, with the effect that these tools consisted of large numbers of partially overlapping and inflexible building blocks.

Parallel to these activities, the focus of research shifted to questions of user interface design. Initially, the purpose of these efforts was to make the previously developed routines easier to apply. The libraries have become quite bulky, and the development of new application programs calling upon one or several of these library routines was a slow process. The efforts were not driven by a desire for improved flexibility. Their only purpose was to protect the users from having to learn how to call the library routines, which often had a formidable list of user-specifiable parameters.

1.3. Historical Background in Second-generation CAD Tools

A major breakthrough was achieved in 1980 with the design of MATLAB. The focus was no longer that of protecting the user from having to understand control algorithms. MATLAB is a very intuitive high-level language for linear matrix algebra. The reason why MATLAB became immediately popular among control engineers was the fact that the previous obscure algorithms became at once very easy to read and understand, to write and enhance, and ultimately maintain.

The potential of the MATLAB approach to control engineering was quickly recognized, and, in the sequel, several second-generation tools were designed on the basis of either the MATLAB software itself or at least the MATLAB language definition. Most of them became commercially available and they will define the state-of-the-art in CAD software.

This new type of tools can be further classified as either comprehensive design tools or design shells. The former type tries to provide algorithms that handle all imaginable control situations. This may result in very large programs offering many different features. The design shells type provides an open-ended operator set that allows the user to code his own algorithms within the frame of the CACSD software.

The goal of a CAD system should be to provide an engineer with a broad spectrum of alternative design possibilities. Now high-performance personal computers promise to spread computing power to even farther reaches. Numerical algorithms combined with the state-of-art user interfaces and graphics may give the control engineer more power at his fingertips than ever before.

2. CAD Techniques

Modern control is based on process models, controller design, simulation and implementation studies. Special purpose CAD systems, which are designed for dedicated tasks in those control disciplines, are not of interest. Those, which allow a combination of the control system design including process modeling and simultaneously the online application of control techniques, are of importance. Rapid control prototyping is the keyword in this context. It is generally referred to as methods and tools to solve control problems fast and efficiently. This includes

- theoretical methods of control engineering from designing a system model and system analysis to the design of a control system,
- test and optimization of the control strategy in a simulation environment,
- automatic code generation for a real-time system operating a test stand and
- verification and optimization of the controller with the target object on the test stand.

The result is a functioning controller prototype. The following steps may be the code optimization for the target controller hardware and the adjustment for application in a series product.

During fast prototyping and the control system design cycle as shown in Figure 1, the problems facing a control engineer are very diverse. Thus he wants to have a set of possible solutions, selecting a special method according to the particular situation and using a large variety of design aids. It seems advantageous to have a comprehensive, easily applicable, extensible and user-friendly software system, which is not only restricted to a special class of problems but also supports all classes of methods and algorithms. The control engineer wants to take tools from a toolbox to handle the diverse problems at the particular stages of design and implementation. In addition to this, he wants to arrange and adapt or modify these tools to special tasks.

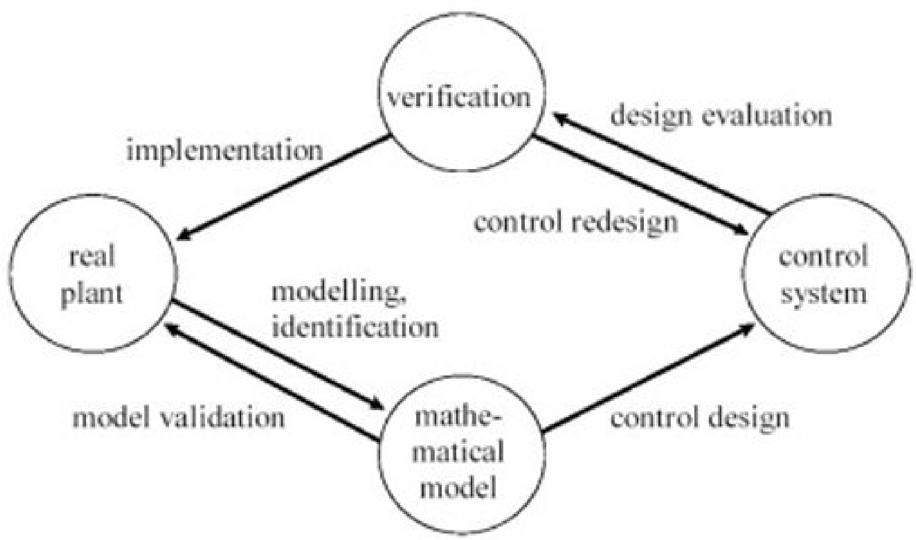

Figure 1: Simplified development cycle of control system design

Most CAD systems try to go beyond theoretical control system analysis to offer support for the task of implementing control laws in hardware and software: interfacing to the real world for data acquisition, control operation and experimental design verification. For developing embedded industrial control devices, special real-time environments are needed. They may be housed in special development systems or by CAD itself, which can be connected directly to the plant to be controlled. Besides performing off-line control analysis and design, a real-time based CAD system must support on-line identification of the plant dynamics, as well as evaluation of prototype control systems. Furthermore, it is advantageous to have a software environment available, where real-time code can be generated and tested on the same computer or downloaded to a process control component. The automatic source-code generation is the essential task in such a system, as currently being followed by MATLAB.

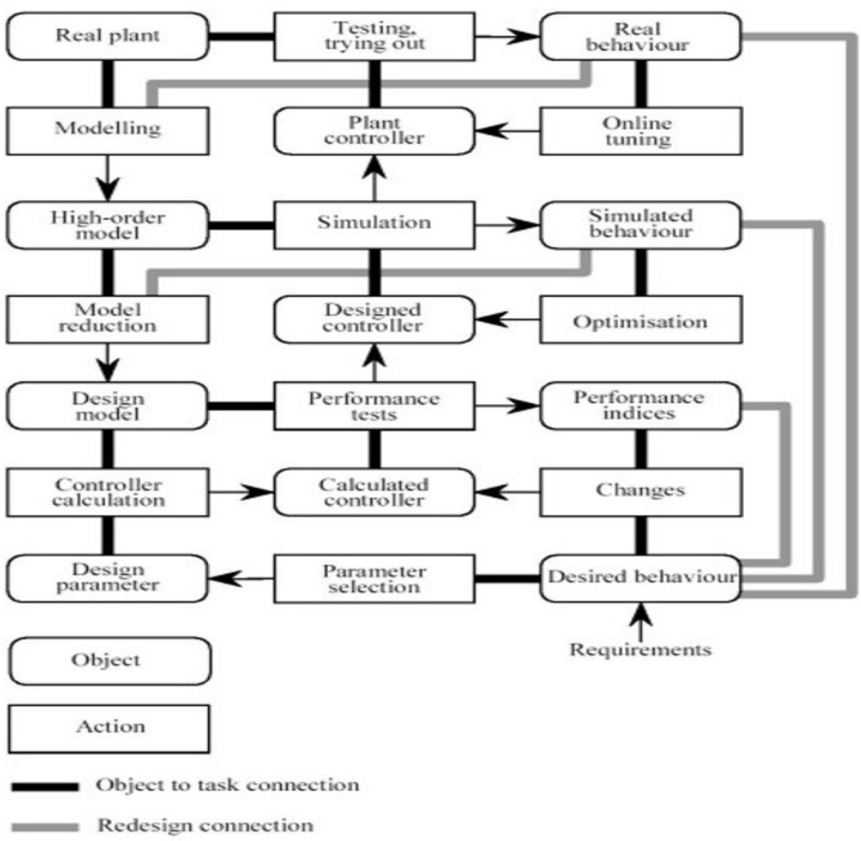

Figure2: Steps performed during design phases of a control system

-
-
-

Bibliography

Anderson E., Bai Z., Bischof C., Blackford S., Demmel J., Dongarra J., Croz J.D., Greenbaum A., Hammarling, S., McKenney A., Sorensen D. (1999). *LAPACK Users' Guide, Third Edition*. Philadelphia: SIAM Press. [A Comprehensive manual and guide for programmers using the current LAPACK subroutine library package].

Benner P., Mehrmann V., Sima V., Van Huffel S., Varga, A. (1999). *SLICOT - A Subroutine Library in Systems and Control Theory*. Vol. 1 (ed. B.N. Datta), pp. 499-539. London: Birkhauser. [Documentation of the 400 user-callable and computational design routines in various domains of systems and control].

Cellier F.E, Rimvall C.M. (1988). Computer-Aided Control System Design: Techniques and Tools. (ed. N.A. Kheir), pp. 631-679. New York: Marcel Dekker. [Overview of CACSD systems, techniques and principles discussed that have become reality].

Elmqvist H., Brück D., Mattsson S.E., Olsson H., Otter M. (2001). *Dymola - Dynamic Modelling Language - User's Manual*. Lund: Dynasim AB. [Introduction into the multidisciplinary modeling system Dymola].

Mann H. (1996). A versatile modelling and simulation tool for mechatronics control system development. *Proceedings of the IEEE International Symposium on CACSD*, pp 524-529. Dearborn. [Introduction into multipole modeling and its application].

Paynter H.M. (1970) System Graphing Concepts. *Instruments Control Systems*, 43(7), pp.77-78. [Introduction into bond graph technique].

The MathWorks Inc. (2000). *Using MATLAB*. Natick. [Comprehensive users manual for MATLAB command language and user interaction].

The MathWorks Inc. (1996). *Control System Toolbox User's Guide*. Natick. [Comprehensive users manual for the MATLAB toolbox dealing with control systems].

The MathWorks Inc. (2000). *Signal Processing Toolbox User's Guide*. Natick. [Comprehensive user manual for the MATLAB toolbox dealing with signals].

The MathWorks Inc. (2002). *Using Simulink*. Natick. [Introducing into modeling and simulation with SIMULINK].

The MathWorks Inc. (2000). *Real-Time Workshop User's Guide*. Natick. [Automatic code generation of system models with SIMULINK].

Tiller M. (2001). *Introduction to physical modelling with Modelica*. Boston: Kluwer Academic Publishers. [A good technical book about this multidisciplinary modeling language].

Biographical Sketch

Christian Schmid received his Diplom Degree in Mechanical Engineering in 1972 from the University of Stuttgart, and a Dr.-Ing. Degree in Electrical Engineering from Ruhr-Universität Bochum in 1979. From 1972 to 1975, he was Research Assistant at the University of Stuttgart. From 1975 to 1992, he was assistant professor and senior lecturer in the Control Engineering Laboratory of Ruhr-Universität Bochum. In 1992 he made the habilitation in Control Engineering. In 1993 he was awarded the title of Privatdozent and in 2000 he was promoted to an apl. Professor of Control Engineering in the Department of Electrical Engineering and Information Sciences at Ruhr-Universität Bochum. His main research interests were numerical methods in control, model reduction, optimization and are currently in the field of computer-aided control system design, adaptive control, applications using neural networks and modeling. He has started projects in the area of multimedia-based learning, virtual and remote labs with German and European partners and is currently managing a nation-wide project on networked learning in the area of control education.